Expressive Completeness of Duration Calculus

Alexander Rabinovich
Department of Computer Science
Sackler Faculty of Exact Sciences
Tel-Aviv University
Ramat Aviv 69978, Israel
e.mail: rabino@math.tau.ac.il

May 2, 2001

Abstract

This paper compares the expressive power of first-order monadic logic of order, a fundamental formalism in mathematical logic and the theory of computation, with that of the Propositional version of Duration Calculus (PDC), a formalism for the specification of real-time systems.

Our results show that the propositional duration calculus is expressively complete for first-order monadic logic of order. Our semantics for PDC conservatively extends the standard one to all positive (including infinite) length intervals. Hence, in view of the expressive completeness, liveness properties can be specified in PDC. This observation refutes a widely believed misconception that the Duration Calculus cannot specify liveness properties.

1 Introduction

The Duration Calculus (DC for brevity) [22] is a formalism for the specification of real time systems. The Duration Calculus is based on interval logic [9] and uses real numbers to model time. DC has been successfully applied in case studies of software embedded systems, e.g., a gas burner [17], a railway crossing [19] and has been used to define the real time semantics of other languages.

A run of a real time system is represented by a function from non-negative reals into a set of values - the instantaneous states of a system. Such a function will be called a signal. Usually, there is a further restriction on the behavior of continuous time systems. For example, a function that assigns value q_0 to the rationals and value q_1 to the irrationals is not accepted as a 'legal' signal.

A requirement that is often imposed in the literature is that in every finite length time interval a system can change its state only finitely many times. This requirement is called finite variability (or non-Zeno) requirement.

Atomic formulas of DC have the form $\lceil S \rceil$, where S is a boolean signal expression. Such a formula has the value true in an interval $[a,\ b]$ if $\int_a^b S$ is equal to $b - a$, i.e., the signal defined by the expression S is true at almost all points of the interval $[a, b]$. If S denotes a finitely variable boolean signal P, then this integral condition is equivalent to "P receives the value false at a finite number of points in the interval $[a,\ b]$."

Note that if P_1 and P_2 are signals that disagree only on a finite number of points in any finite length interval $[c,\ d]$ (notation $P_1 \sim_{fin} P_2$), then $\int_a^b P_1 = \int_a^b P_2$. The Duration Calculus formulas *respect* $\sim_{fin}$ equivalence, i.e., if $P_1 \sim_{fin} P_2$, then P_1 satisfies a duration formula D if and only if P_2 satisfies D. Therefore, in DC it is impossible to specify instantaneous events. In [24], DC was extended to Mean Value Calculus in order to handle instantaneous events.

We investigate the expressive power of the Propositional fragment of Duration Calculus (PDC). In this fragment the metric aspects of the calculus are ignored. Hence, this is a duration free fragment of the Duration Calculus. Our aim is to clarify the logical foundation of DC. A justification for the significance of the Propositional (duration free) fragment of DC is essential.

- Though the metrical aspects of the Duration Calculus are very important in applications, their investigation may require incursions into Calculus (e.g., into differential equations) which have little (if anything) in common with existing well-understood tools of Logic and computational model theory.

- The duration free fragment of the Duration Calculus plays a fundamental role in applications. In fact, the majority of the laws and the transformation rules in [16] deal with non-metric aspects of the duration calculus.

- The duration free fragment is central in reasoning about the Duration and Mean Value Calculi. For example, in [24], nine out of ten axioms for the Mean Value Calculus are duration free (non-metrical).

The main result of this paper is the following equivalence between Propositional Duration Calculus and first-order monadic logic of order.

Theorem (Expressive completeness of PDC for first-order monadic logic of order)

1. Every PDC formula is equivalent to a monadic sentence which respects $\sim_{fin}$ equivalence.

2. Every monadic sentence which respects $\sim_{fin}$ equivalence is equivalent to a PDC formula.

This result confirms that the Duration Calculus is not an ad-hoc formalism because it is so tightly related to first-order monadic logic of order, a fundamental formalism in mathematical logic and the theory of computation.

It is a widely believed misconception that the Duration Calculus cannot specify liveness properties while monadic first-order logic can. Recently, several extensions of DC were suggested (see e.g., [10, 18]) for the specification of liveness properties in DC. All these extensions can be easily formalized in first-order monadic logic (in the fragment that respects $\sim_{fin}$-equivalence), hence, in view of the above theorem they do not increase the expressive power of the Duration Calculus. Some reasons for this misconception are explained in Remark 18 (Section 4) and in Section 8.

The rest of the paper is organized as follows. In Section 2 notations and terminology are explained. In Section 3 the syntax and the semantics of monadic logic are recalled. In Section 4 the syntax and the semantics of duration calculus are provided. Our definitions conservatively extend the traditional semantics of DC to all (including infinite length) subintervals of the reals. Section 5 states the expressive completeness theorem. The proof of the theorem is given in Section 6 and Section 7. In Section 8 we compare the traditional extensions of the duration calculus to infinite intervals with our extension. We show that liveness properties cannot be specified under the traditional extensions and that unlike our extensions, the traditional extensions are not expressively complete. In Section 9 we provide a novel stuttering free interpretation for star-free expressions and establish the equivalence between star-free expressions under stuttering free interpretation and PDC. Section 10 describes some related results.

2 Notations

$\mathbb{N}$ is the set of natural numbers; $\mathbb{BOOL}$ is the set of booleans, $\mathbb{R}$ is the set of real numbers, $\mathbb{R}^{\geq 0}$ is the set of non-negative reals; $a,\ b$ will range over $\mathbb{R}^{\geq 0}$; $[a,\ b]$ is a finite length closed interval on the reals; we will use the standard notations for other types of intervals, e.g., $(a,\ b)$ is an open interval; all intervals are assumed to be non-empty sets; an interval is singular if it contains only one point; I will range over intervals.

A monadic predicate over I is a function from I into $\mathbb{BOOL}$; the letters P_i range over monadic predicates. Whenever the domain I of a monadic predicate is clear from the context we use 'predicate' for 'monadic predicate over I'. As usual, a subset of a set I will be identified with its characteristic predicate.

For a predicate P over I and a subset I_1 of I we denote by $P \restriction I_1$ the restriction of P on I_1.

3 Monadic First Order Logic of Order

In the first subsection we briefly recall the syntax and the semantics of first-order monadic logic of order. The standard syntactical extension of monadic logic by bounded quantifiers is given in the second subsection. Some lemmas which are referred to later are also stated there.

3.1 Syntax and Semantics

First-order monadic logic of order is a fundamental formalism in mathematical logic and the theory of computation. We use $X_1, \ldots, X_n$ for monadic predicate symbols and $t,\ u,\ v$ for first order variables. The *atomic* formulas are formulas of the form $X_i(t)$ and $t < u$. The formulas are constructed from the atomic formulas by the propositional connectives and the first order quantifiers.

Free and bound variables are defined as usual. We will use the notation $\phi\{u/v\}$ for the formula obtained from ϕ by replacing all free occurrences of v by u and renaming bound variables, if necessary. If all free variables of ϕ are among $\{t_1, \ldots, t_n\}$, we write $\phi(t_1, \ldots, t_n)$. Recall that a sentence is a formula without free variables. The quantifier depth of a formula is also defined in the standard way (see e.g. [8]).

A structure $K = \langle I_K,\ <_K,\ P_1^K, \ldots, P_n^K \rangle$ for monadic logic of order consists of a set I_K linearly ordered by $<_K$ and subsets P_i^K of I_K.

The notion of satisfaction of a formula in a structure is defined as usual. We write $(K, a_1, \ldots, a_n) \models \phi(t_1, \ldots, t_n)$ or $K \models \phi(a_1, \ldots, a_n)$ if $\phi(t_1, \ldots, t_n)$ holds in K, whenever the variables $t_1, \ldots, t_n$ are interpreted as the elements $a_1, \ldots, a_n$ of I_K.

Notation Let $K = \langle I_K,\ <_K,\ P_1^K, \ldots, P_n^K \rangle$ be a structure and let I be a subset of I_K. We use $K \restriction I$ for the restriction of K on I; this is the structure $\langle I,\ <_K \cap I \times I,\ \ P_1^K \restriction I,\ \ldots P_n^K \restriction I \rangle$. In particular, we use $K \restriction [a,\ b]$ for the restriction of K on $[a,\ b]$. We use $K \restriction \geq a$ for the restriction of K on the interval $\{c \in I_K\ : c \geq_K a\}$. The structures $K \restriction \leq a$, $K \restriction < a$ and $K \restriction > a$ are defined similarly.

3.2 Extension by Bounded Quantifiers

It is convenient to extend the syntax of first-order monadic logic of order by the bounded quantifiers $(\exists t)_{t_1}^{t_2}$, $(\forall t)_{t_1}^{t_2}$, $(\exists t)_{t_1}$, $(\forall t)_{t_1}$, $(\exists t)^{t_2}$, $(\forall t)^{t_2}$. Semantically, $(\exists t)_{t_1}^{t_2}\phi$ and $(\forall t)_{t_1}^{t_2}\phi$ are shorthand for $\exists t.\ t_1 < t < t_2 \wedge \phi$ and $\forall t.\ t_1 < t < t_2 \rightarrow \phi$. The variable t_1 (respectively t_2) is called the lower (respectively the upper) limit of the quantifiers $(\exists t)_{t_1}^{t_2}$ and $(\forall t)_{t_1}^{t_2}$. Both t_1 and t_2 are free in $(\exists t)_{t_1}^{t_2}\phi$. The meaning of the quantifiers $(\exists t)_{t_1}$, $(\forall t)_{t_1}$, $(\exists t)^{t_2}$, $(\forall t)^{t_2}$ is defined similarly, e.g. $(\exists t)_{t_1}\phi$ is a shorthand for $\exists t.\ t_1 < t \wedge \phi$. The quantifiers $(\exists t)_{t_1}$, $(\forall t)_{t_1}$ have the lower limit t_1 and do not have an upper limit. The quantifiers $(\exists t)^{t_2}$, $(\forall t)^{t_2}$ have the upper limit t_2 and do not have a lower limit.

Let ϕ be a formula with only bounded quantifiers (without loss of generality we assume that each variable name is bound at most once in ϕ). A sequence $t_1, \ldots, t_n$ is called a *lower* (*upper*) sequence of ϕ if (1) t_1 is a bound variable of ϕ; (2) t_{i+1} is the lower (respectively, the upper) limit of the quantifier that binds t_i and (3) t_n is free in ϕ.

Example 1 *$X(t_1) \wedge t_1 < t_2$ does not have any lower and any upper sequences. The lower (respectively upper) sequences of $X(t_1) \wedge (\exists v)_u^w(X(v) \wedge \neg(\exists t_2)_{t_1}^v X(t_2))$ are t_2, t_1 and v, u (respectively t_2, v, w and v, w).*

A formula is said to be *explicitly restricted* to $[t_1, t_2]$ if (1) all the quantifiers of the formula have a lower and an upper limit, (2) the set of its free variables is a subset of $\{t_1,\ t_2\}$ and (3) every lower sequence of the formula ends with t_1 and every upper sequence ends with t_2. We say that $\phi(t_1,\ t_2)$ is explicitly

restricted to $(t_1, t_2]$ (respectively, $[t_1,\ t_2)$, or respectively $(t_1,\ t_2)$) if ϕ is explicitly restricted to $[t_1, t_2]$ and it does not contain an occurrence of $X(t_1)$ (respectively, $X(t_2)$, or respectively $X(t_1)$ and $X(t_2)$).

Example 2 *If all the quantifiers of $\phi(t_1,\ t_2)$ are relativized to $(t_1,\ t_2)$, i.e., have the form $(\exists v)^{t_2}_{t_1}$, then $\phi(t_1,\ t_2)$ is explicitly restricted to $[t_1,\ t_2]$. Note that according to our definition the formula $t_1 < t \wedge t < t_2$ is not explicitly restricted to $[t_1,\ t_2]$.*

A formula is said to be *explicitly restricted* from above by t_1 if (1) all the quantifiers of the formula have an upper limit, (2) t_1 is the only free variable of the formula and (3) every upper sequence ends with t_1. If, in addition, the formula does not contain occurrences of sub-formulas of the form $X(t_1)$, then we say that it is explicitly restricted from above by t_1^-. A formula is *explicitly restricted* from below by t_1 if (1) all the quantifiers of the formula have a lower limit, (2) t_1 is the only free variable of the formula and (3) every lower sequence ends with t_1. If, in addition, the formula does not contain occurrences of sub-formulas of the form $X(t_1)$, then we say that it is explicitly restricted from below by t_1^+.

The following two lemmas are straightforward.

Lemma 3 *Let $K = \langle I_K,\ <_K,\ P_1^K, \ldots, P_n^K\rangle$ be a structure and let $a_1 \leq a_2$ be elements of I_K.*

1. *If $\phi(t_1,\ t_2)$ is explicitly restricted to $[t_1,\ t_2]$, then $K \models \phi(a_1,\ a_2)$ if and only if $K \restriction [a_1,\ a_2] \models \phi(a_1,\ a_2)$.*

2. *If $\phi(t_1)$ is explicitly restricted from below by t_1 then, $K \models \phi(a_1)$ if and only if $K \restriction \geq a_1 \models \phi(a_1)$.*

Hence, if K and K' coincide on $[a, b] \subseteq I_K \cap I_{K'}$ and $\phi(t_1, t_2)$ is explicitly restricted to $[t_1, t_2]$, then $K \models \phi(a_1,\ a_2)$ if and only if $K' \models \phi(a_1,\ a_2)$.

Lemma 4 *Let $\phi(t_1,\ t_2)$ be a formula explicitly restricted to $[t_1,\ t_2)$ and $\chi(t_1)$ be a formula obtained from ϕ as follows: (1) Remove t_2 from the upper limits of all quantifiers; (2) Replace the sub-formulas $v < t_2$ by $\neg t_1 < t_1$ (i.e. by a true statement) and (3) Replace the sub-formulas $t_2 < u$ and $t_2 < t_2$ by $t_1 < t_1$ (i.e. by a false statement). Then $K \models \phi(a_1,\ a_2)$ iff $K \restriction [a_1, a_2) \models \chi(a_1)$.*

Lemma 5 *([1] Lemma 9.3.2) There exists an algorithm that for every formula $\phi(t_1,\ t_2)$ constructs a formula $\psi(t_1,\ t_2)$ of the form $\bigvee_{i \in Ind}(\psi^i_<(t_1) \wedge \psi^i_b(t_1) \wedge \psi^i(t_1,\ t_2) \wedge \psi^i_e(t_2) \wedge \psi^i_>(t_2))$, where Ind is a finite set, such that*

$$t_1 < t_2 \wedge \phi(t_1,\ t_2) \text{ is equivalent to } t_1 < t_2 \wedge \psi(t_1,\ t_2)$$

and for all $i \in Ind$

1. ψ^i_b *and* ψ^i_e *are quantifier free.*

2. $\psi^i(t_1,\ t_2)$ *is explicitly restricted to* $(t_1,\ t_2)$.

3. $\psi^i_<(t_1)$ *is explicitly restricted from above by* t_1^-, *and* $\psi^i_>(t_2)$ *is explicitly restricted from below by* t_2^+.

4. *The quantifier depths of* $\psi^i(t_1,\ t_2)$, $\psi^i_<(t_1)$ *and* $\psi^i_>(t_2)$ *are at most the quantifier depth of* $\phi(t_1,\ \ t_2)$.

Remark 6 *The assertion that* ψ *is obtained from* ϕ *by an algorithm, and that the quantifier depths of the components* ψ^i *etc. are at most that of* ϕ *are corollaries of the proof of Lemma 9.3.2 in [1], not its statement.*

As a consequence of Lemma 5 we obtain

Lemma 7 *There exists an algorithm that for every formula* $\phi(t_1,\ t_2)$ *constructs a formula* $\psi(t_1,\ t_2)$ *of the form* $\bigvee_i(\psi^i(t_1,\ t_2) \wedge \psi^i_\geq(t_2))$ *such that*

1. *The quantifier depths of* $\psi^i(t_1,\ t_2)$ *and* $\psi^i_\geq(t_1)$ *are at most the quantifier depth of* $\phi(t_1,\ \ t_2)$.

2. $\psi^i(t_1,\ t_2)$ *is explicitly restricted to* $[t_1,\ t_2)$ *and* $\psi^i_\geq(t_2)$ *is explicitly restricted by* t_2 *from below.*

3. *For every structure* K, *if* min_K *is the least element of* I_K *and* $min_K < e \in I_K$, *then* $K \models \phi(min_K,\ e)$ *iff* $K \models \psi(min_K, e)$.

Proof. The lemma follows from Lemma 5 and the observation that if a formula $\chi(t)$ is explicitly restricted from above by t^-, then either (1) or (2) below holds: (1) for every structure K, if min_K is the minimal element of I_K, then $K \models \chi(min_K)$; (2) for every structure K, if min_K is the minimal element of I_K, then $K \not\models \chi(min_K)$. Moreover, it is decidable which of the cases holds.

Let us explain how to decide which of the above cases holds. First observe that if a formula $\chi(t)$ is explicitly restricted from above by t^-, then $\chi(t)$ is a boolean combination of formulas of the following form: $(\exists t')^t\phi_i$, $(\forall t')^t\phi_j$ and $t < t$. Now replace the sub-formulas of the form "$(\exists t')^t\phi_i$" and "$t < t$" by *FALSE* and the sub-formulas of the form "$(\forall t')^t\phi_j$" by *TRUE*. If the result is *TRUE*, then case (1) holds, otherwise case (2) holds. □

4 Propositional Duration Calculus

The Duration Calculus (DC) [22] is a formalism for the specification of real time systems. DC is based on interval logic [9] and uses real numbers to model time. DC has been successfully applied in case studies of software embedded systems, e.g., a gas burner [17], a railway crossing [19] and has been used to define the real time semantics of other languages.

The Propositional Duration Calculus (PDC) (called the restricted duration calculus in [3]) is a fragment of the duration calculus.

In this section the syntax and the semantics of PDC are presented. The definition of the semantics differs from the traditional definition (see e.g. [3]). The main difference is that we provide the semantics over all positive length intervals (including infinite intervals), whereas traditionally the semantics is defined only for the intervals of the form $[a,\ b]$. Our semantics conservatively extends the traditional semantics. The detailed explanation of the differences between the definitions given here and the traditional definitions is provided in Remark 16 of Section 4.3.

4.1 Syntax

The sets of formulas of PDC are parameterized by a set $X_1, \ldots, X_n$ of state variables that 'correspond' to the monadic predicates of first-order logic.

PDC has two syntactical categories: state expressions and formulas.

State Expressions: The *state expressions* are constructed from the state variables by propositional connectives. We will use S to range over the state expressions which are defined by the following grammar:

$$S ::= X \mid S \vee S \mid S \wedge S \mid \neg S, \text{ where } X \text{ is a state variable.}$$

Atomic Formulas of PDC: if S is a state expression, then $\lceil S \rceil$ is an atomic formula of PDC.

Formulas: The formulas of PDC are defined by the following grammar:

$$D ::= \ At \mid D^\frown D \mid \neg D \mid D \vee D \mid D \wedge D, \text{ where } At \text{ ranges over the atomic formulas.}$$

The binary operation $^\frown$ is called "chop".

4.2 Semantics of PDC

A run of a real time system is represented by a function from non-negative reals into a set of values - the instantaneous states of a system. Such a function will be called a signal. Usually, there is a further restriction on the behavior of continuous time systems. For example, a function that assigns value q_0 to the rationals and value q_1 to the irrationals is not accepted as a 'legal' signal.

A requirement that is often imposed in the literature is that in every finite length time interval a system can change its state only finitely many times. This requirement is called finite variability (or non-Zeno) requirement.

Definition 8 *A predicate P over $\mathbb{R}^{\geq 0}$ has finite variability if there exists an unbounded increasing sequence $a_0 = 0 < a_1 < a_2 \ldots < a_n < \ldots$ such that P is constant on every interval (a_i, a_{i+1}). A predicate P over a subset I of $\mathbb{R}^{\geq 0}$ has finite variability if P is the restriction on I of a finitely variable predicate.*

In the literature finitely variable predicates are sometimes called non-Zeno boolean signals or piecewise continuous trajectories.

Let us first motivate the semantical definitions. The state variables are interpreted as finitely variable predicates. Atomic formulas of the form $\lceil X_i \rceil$, where X_i is a state variable has the value true in a finite length interval I if $\int_I X_i$ is equal to the length of I, i.e., the predicate P_i assigned to X_i is true at almost all points of the interval I. If P_i is a finitely variable predicate over a finite length interval I, then this integral condition is equivalent to "P_i receives the value true at all but finitely many points in the interval I." It is clear that the truth value of $\lceil X_i \rceil$ will be invariant under the changing of the value of P_i at a finite number of points or by adding/deleting endpoints to the interval I. Formulas that have this invariance property are said to respect $\sim_{fin}$-equivalence (for the formal definition see Definition 13). All Duration Calculus formulas respect $\sim_{fin}$-equivalence.

Now let us proceed with formal definitions.

A structure $K = \langle I_K, <_K, P_1^K, \ldots, P_n^K \rangle$ is a *finitely variable* structure if I_K is a non-singular subinterval of $\mathbb{R}^{\geq 0}$ and for $i = 1, \ldots, n$ the predicates P_i^K are finitely variable predicates over I_K. Recall that an interval is singular if it contains only one point, hence a non-singular interval has a positive (including infinite) length.

The PDC formulas and expressions are interpreted in the finitely variable structures.

Let $K = \langle I_K, <_K, P_1^K, \ldots, P_n^K \rangle$ be a finitely variable structure. The meaning (in K) of state expressions and formulas is provided below.

State Expressions: The meaning $[\![S]\!]^K$ of a state expression S in a structure K is a subset of I_K defined as usual by structural induction on the state expressions. Namely,

Variables: $[\![X_i]\!]^K$ is P_i^K.

Disjunction: $[\![S_1 \vee S_2]\!]^K$ is the union of $[\![S_1]\!]^K$ and $[\![S_2]\!]^K$.

Conjunction: $[\![S_1 \wedge S_2]\!]^K$ is the intersection of $[\![S_1]\!]^K$ and $[\![S_2]\!]^K$.

Negation: $[\![\neg S]\!]^K$ is the complement of $[\![S]\!]^K$ relative to I_K.

It is easy to check that if K is a finitely variable structure, then the characteristic function of $[\![S]\!]^K$ has finite variability.

When K is clear from the context we will use $[\![S]\!]$ for $[\![S]\!]^K$; furthermore, we sometimes identify the set $[\![S]\!]$ with its characteristic function.

The satisfaction relation between PDC formulas and the finitely variable structures is defined as follows.

PDC Atomic Formulas: $K \models \lceil S \rceil$ if the complement of $[\![S]\!]^K$ relative to I_K does not contain an interval of a positive length.

Since $[\![S]\!]^K$ has finite variability, the above requirement is equivalent to the requirement that $[\![S]\!]^K$ has the value false at a finite number of points in every finite length subinterval of I_K. For a bounded non-singular interval I_K this condition is equivalent to the requirement that $\int_{I_K} [\![S]\!]^K$ is equal to the length of I_K.

The meanings of the propositional connectives are defined as usual: (1) $K \models D_1 \vee D_2$ if $K \models D_1$ or $K \models D_2$; (2) $K \models D_1 \wedge D_2$ if $K \models D_1$ and $K \models D_2$; (3) $K \models \neg D$ if not $K \models D$.

Recall that every interval is a non-empty set and an interval I_1 precedes an interval I_2 if $a \in I_1 \wedge b \in I_2 \rightarrow a < b$. A *chop-partition* of an interval I is an ordered pair of intervals I_1 and I_2 such that $I = I_1 \cup I_2$ and I_1 precedes I_2. Note that the requirement that I_1 precedes I_2 implies that the intervals I_1 and I_2 are disjoint. Here are some chop-partitions for the interval $I = [0,\ 1)$: (1) $I_1 = [0,\ \frac{1}{4}]$, $I_2 = (\frac{1}{4},\ 1)$; (2) $I_1 = [0,\ \frac{1}{4})$, $I_2 = [\frac{1}{4},\ 1)$ and (3) $I_1 = \{0\}$, $I_2 = (0,\ 1)$. Observe that a one-point interval has no chop-partition.

Now we are ready to define the semantics of chop.

Chop: $K \models D_1{}^\frown D_2$ if and only if $K \restriction I_1 \models D_1$ and $K \restriction I_2 \models D_2$ for some chop-partition of I_K into positive length subintervals $I_1,\ I_2$.

4.3 PDC respects $\sim_{fin}$-equivalence

Throughout this section all structures are assumed to be finitely variable.

Definition 9 *Predicates P_1 and P_2 are said to be $\sim_{fin}$-equivalent on a subinterval I of $\mathbb{R}^{\geq 0}$ iff for every $b \in \mathbb{R}^{\geq 0}$ the set $\{c \in I \ : c < b \wedge P_1(c) \neq P_2(c)\}$ is finite. Predicates over an interval I are said to be $\sim_{fin}$-equivalent if they are $\sim_{fin}$-equivalent on I.*

The following lemmas are easily proved by structural induction on PDC formulas.

Lemma 10 *Let $K = \langle I, P_1^K, \ldots P_n^K \rangle$ and $K' = \langle I, P_1^{K'}, \ldots P_n^{K'} \rangle$ be two structures over the same interval I. Suppose that $P_i^K \sim_{fin} P_i^{K'}$ for $i = 1, \ldots n$. Let D be any PDC formula. Then $K \models D$ if and only if $K' \models D$.*

Recall that a point a is an internal point of an interval I if there exists $\epsilon > 0$ such that the interval $(a - \epsilon,\ a + \epsilon)$ is a subset of I.

Lemma 11 *Let $K = \langle I_K, P_1^K, \ldots P_n^K \rangle$ and $K' = \langle I_{K'}, P_1^{K'}, \ldots P_n^{K'} \rangle$ be two structures. Suppose that I_K and $I_{K'}$ have the same set I^o of internal points and that $P_i^K \restriction I^o = P_i^{K'} \restriction I^o$ for $i = 1, \ldots n$. Then $K \models D$ if and only if $K' \models D$ for every PDC formula D.*

The $\sim_{fin}$ equivalence on predicates is extended to the structures as follows:

Definition 12 *($\sim_{fin}$ equivalence on structures) Structures $K = \langle I_K, P_1^K, \ldots P_n^K \rangle$ and $K' = \langle I_{K'}, P_1^{K'}, \ldots P_n^{K'} \rangle$ are $\sim_{fin}$ equivalent (notation $K \sim_{fin} K'$) if*

1. *I_K and $I_{K'}$ have the same set I^o of internal points.*
2. *$P_i^K \restriction I^o$ is $\sim_{fin}$-equivalent to $P_i^{K'} \restriction I^o$, for $i = 1, \ldots, n$.*

Observe that if $K \sim_{fin} K'$ and K is a finitely variable structure, then K' is finitely variable.

Definition 13 *A formula D respects $\sim_{fin}$ equivalence if $K \sim_{fin} K'$ and K is finitely variable imply that $K \models D$ if and only if $K' \models D$.*

Lemma 10 and Lemma 11 imply

Proposition 14 *The PDC formulas respect $\sim_{fin}$ equivalence.*

The following proposition provides alternative definitions for the chop operator of the Duration Calculus.

Proposition 15 *For PDC formulas D_1 and D_2 the following conditions are equivalent.*

1. $K \models D_1^\frown D_2$
2. *There is an internal point b of the interval I_K such that $K \restriction \leq b \models D_1$ and $K \restriction \geq b \models D_2$.*
3. *There is an internal point b of the interval I_K such that $K \restriction \leq b \models D_1$ and $K \restriction > b \models D_2$.*
4. *There is an internal point b of the interval I_K such that $K \restriction < b \models D_1$ and $K \restriction \geq b \models D_2$.*
5. *There is an internal point b of the interval I_K such that $K \restriction < b \models D_1$ and $K \restriction > b \models D_2$.*

Proof. Immediately from Proposition 14. □

Remark 16 *(Comparison with the traditional semantics of PDC [3].) Let us point to the following differences in the semantical definitions:*

1. *(*Intervals.*) We provided the semantics of PDC formulas over all positive length (including infinite) intervals, whereas traditionally the semantics is given only for the intervals of the form $[a,\ b]$ ($a < b \in \mathbb{R}^{\geq 0}$).*
2. *(*Chop.*) Consequently, there is a discrepancy in the definition of chop. Namely, in the traditional versions $\langle [a,\ b],\ <, P_1, \ldots P_n\rangle \models D_1^\frown D_2$ if and only if $\langle [a,\ m],\ <,\ P_1, \ldots P_n\rangle \models D_1$ and $\langle [m,\ b],\ <,\ P_1, \ldots P_n\rangle \models D_2$ for some $m \in (a,\ b)$.*

In view of Proposition 15 the difference in the definitions of chop for PDC is not essential. Hence, our definitions conservatively extend the traditional semantics of PDC to arbitrary positive (including infinite) length intervals.

5 Expressive Completeness

A formula ψ_1 is equivalent to a formula ψ_2 over a set of structures CL, if $K \models \psi_1 \iff K \models \psi_2$ for every structure $K \in CL$.

We say that ψ_1 is equivalent to ψ_2 if ψ_1 is equivalent to ψ_2 over the set of finitely variable structures.

The main result of this paper is the following equivalence between Propositional Duration Calculus and first-order monadic logic of order.

Theorem 17 *(Expressive Completeness)*

1. *There exists a translation algorithm Tr, assigning to a PDC formula D a first-order monadic sentence $Tr(D)$ that respects $\sim_{fin}$-equivalence such that D is equivalent to $Tr(D)$.*

2. *There exists a translation algorithm Tr', assigning to a first-order monadic sentence ψ a PDC formula $Tr'(\psi)$ such that if ψ respects $\sim_{fin}$-equivalence, then ψ is equivalent to $Tr'(\psi)$.*

The proof of the theorem is provided in Section 6 and Section 7.

Remark 18 *It is a widely believed misconception that the Duration Calculus cannot specify liveness properties while monadic first-order logic can. Recently, several extensions of DC were suggested (see e.g., [10, 18]) for the specification of liveness properties in DC. All these extensions can be easily formalized in the fragment of first-order monadic logic that respects $\sim_{fin}$-equivalence; hence, in view of the above theorem they do not increase the expressive power of the Duration Calculus. The following points are the sources of this misconception:*

- *The "traditional" semantics of DC is defined only on the finite length closed subintervals of the reals. Clearly, there are no liveness properties over bounded time domains. The semantics given in this paper conservatively extends the traditional semantics to all intervals.*

- *The extensions of DC to infinite intervals considered in the literature (see e.g. [25]) use a convention that a formula holds on an infinite interval if it holds on all its finite length prefixes. Clearly, if such a convention is used, only safety properties can be expressed.*

In Section 8 we compare the traditional extensions of the duration calculus to infinite intervals with our extension.

6 Proof of Theorem 17(1)

Let us fix an individual variable v. For a PDC expression S we denote by $exp(S)$ the monadic formula which is obtained from S, when all occurrences of the state variables X_i are replaced by the monadic formulas $X_i(v)$. In Fig. 1 four translations Tr^{bound}, Tr^{ubound}, Tr^{lbound}, Tr from the Duration formulas into monadic logic are defined. In this definition $T \in \{Tr^{bound}, Tr^{ubound}, Tr^{lbound}, Tr\}$.

First, some comments about the translations and their properties are provided. Later, we state lemmas which establish the correctness of the translations.

Let $K = \langle I_K, <_K, P_1^K, \ldots, P_n^K \rangle$ be a finitely variable structure where I_K is an interval $[b, e]$. Let S be a state expression.

Note that a boolean combination of finitely variable signals is a finitely variable signal. Therefore, $[|S|]^K$ can change only a finite number of times in a bounded interval. Hence, $[|S|]^K$ is true almost everywhere in $[b, e]$ (i.e. the complement of $[\![S]\!]^K$ relative to I_K does not contain an interval of a positive length) iff for every m in the interval (b, e) the signal defined by the state expression S is true in an open non-empty subinterval with m as its left endpoint. $Tr^{bound}(\lceil S \rceil)$ just formalizes this condition. Namely, $Tr^{bound}(\lceil S \rceil)$ is the formula $\phi(t_1, t_2)$ which says that every t in the interval (t_1, t_2) is the left endpoint of an open nonempty subinterval (t, t') where S holds.

Therefore, $K \models \lceil S \rceil$ iff $(K, b, e) \models \phi(t_1, t_2)$.

Observe that $\phi(t_1, t_2) \triangleq Tr^{bound}(\lceil S \rceil)$ is explicitly restricted to (t_1, t_2).

Hence, by Lemma 3, $\lceil S \rceil$ and its Tr^{bound} translation $\phi(t_1, t_2)$ are related as follows:

$$\text{if } K' = \langle I_{K'}, P_1^{K'}, \ldots P_n^{K'} \rangle \text{ and } b < e \text{ and } [b, e] \subseteq I_{K'} \text{ then}$$

$$K' \restriction [b, e] \models \lceil S \rceil \text{ iff } (K', b, e) \models \phi(t_1, t_2).$$

The same relation holds between an arbitrary PDC formula D and its Tr^{bound} translation $\phi(t_1, t_2)$. Namely, if $K' = \langle I_{K'}, P_1^{K'}, \ldots P_n^{K'} \rangle$ and $b < e \in I_{K'}$, then $K' \restriction [b, e] \models D$ iff $(K', b, e) \models \phi(t_1, t_2)$. To establish such a relation one can proceed by induction on PDC formulas. The inductive steps for boolean connectives are trivial and the inductive step for chop is easily obtained from the semantical definition of chop and Proposition 15.

The translations Tr^{lbound} and Tr^{ubound} have similar properties. The only free variable of $Tr^{lbound}(D)$ (respectively $Tr^{ubound}(D)$) is t_1 (respectively t_2).

$$
\begin{array}{lcl}
Tr^{bound}(\lceil S \rceil) & = & (\forall t)_{t_1}^{t_2}(\exists t')_t^{t_2}(\forall t'')_t^{t'} exp(S)\{t''/v\} \\
Tr^{bound}(D_1^\frown D_2) & = & (\exists t)_{t_1}^{t_2} Tr^{bound}(D_1)\{t/t_2\} \wedge Tr^{bound}(D_2)\{t/t_1\} \\
Tr^{lbound}(\lceil S \rceil) & = & (\forall t)_{t_1}(\exists t')_{t_1}^{t}(\forall t'')_{t'}^{t} exp(S)\{t''/v\} \\
Tr^{lbound}(D_1^\frown D_2) & = & (\exists t)_{t_1} Tr^{bound}(D_1)\{t/t_2\} \wedge Tr^{lbound}(D_2)\{t/t_1\} \\
Tr^{ubound}(\lceil S \rceil) & = & (\forall t)^{t_2}(\exists t')_t^{t_2}(\forall t'')_t^{t'} exp(S)\{t''/v\} \\
Tr^{ubound}(D_1^\frown D_2) & = & (\exists t)^{t_2} Tr^{ubound}(D_1)\{t/t_2\} \wedge Tr^{bound}(D_2)\{t/t_1\} \\
Tr(\lceil S \rceil) & = & (\forall t)(\exists t'.t' > t \rightarrow (\exists t')_t(\forall t'')_t^{t'} exp(S)\{t''/v\} \\
Tr(D_1^\frown D_2) & = & (\exists tt't''.t' < t < t'' \wedge Tr^{ubound}(D_1)\{t/t_2\} \wedge Tr^{lbound}(D_2)\{t/t_1\} \\
T(D_1\ op\ D_2) & = & T(D_1)\ op\ T(D_2), \text{ where } op \in \{\wedge,\ \vee\}. \\
T(\neg D) & = & \neg T(D).
\end{array}
$$

Figure 1: Translations

Let $\phi_l(t_1)$ be $Tr^{lbound}(D)$ and $\phi_u(t_2)$ be $Tr^{ubound}(D)$. Let $K' = \langle I_{K'}, P_1^{K'}, \ldots P_n^{K'} \rangle$ be a finitely variable structure and assume that $a_1,\ a_2$ are internal points of $I_{K'}$. Then (1) $K' \restriction \geq a_1 \models D$ iff $(K', a_1) \models \phi_l(t_1)$ and (2) $K' \restriction \leq a_2 \models D$ iff $(K', a_2) \models \phi_u(t_2)$

Now let us state lemmas which establish the correctness of our translations. The proofs of these lemmas are straightforward.

Lemma 19 *(Syntactical properties of the translations)*

1. *Tr^{bound} maps PDC formulas to monadic formulas which have two free variables $t_1,\ t_2$ and are explicitly restricted to $(t_1,\ t_2)$.*

2. *Tr^{lbound} maps PDC formulas to monadic formulas which have one free variable t_1 and are explicitly restricted from below by t_1^+.*

3. *Tr^{ubound} maps PDC formulas to monadic formulas which have one free variable t_2 and are explicitly restricted from above by t_2^-.*

4. *Tr maps PDC formulas to monadic sentences.*

Lemma 20 *(Translations respect $\sim_{fin}$-equivalence.) Let $K = \langle I_K, P_1^K, \ldots P_n^K \rangle$ and $K' = \langle I_{K'}, P_1^{K'}, \ldots P_n^{K'} \rangle$ be finitely variable structures. Assume that I_K and $I_{K'}$ have the same set of internal points and that $a, b \in I_K \cap I_{K'}$.*

1. *If P_i^K is $\sim_{fin}$-equivalent to $P_i^{K'}$ on $(a,\ b)$ for $i = 1, \ldots, n$, then $(K,\ a,\ b) \models Tr^{bound}(D)$ iff $(K',\ a,\ b) \models Tr^{bound}(D)$.*

2. *If* P_i^K *is* $\sim_{fin}$*-equivalent to* $P_i^{K'}$ *on* $\{c \in I \ : \ c < a\}$ *for* $i = 1, \ldots, n$, *then* $(K,\ a) \models Tr^{ubound}(D)$ *iff* $(K',\ a) \models Tr^{ubound}(D)$.

3. *If* P_i^K *is* $\sim_{fin}$*-equivalent to* $P_i^{K'}$ *on* $\{c \in I \ : \ c > a\}$ $i = 1, \ldots, n$, *then* $(K,\ a) \models Tr^{lbound}(D)$ *iff* $(K',\ a) \models Tr^{lbound}(D)$.

4. $Tr(D)$ *respects* $\sim_{fin}$*-equivalence.*

Lemma 21 *Let* K *be a finitely variable structure.*

1. *Assume that* $a < b$ *and* $[a, b] \subseteq I_K$. *Then* $K \restriction [a, b] \models D$ *iff* $(K, a,\ b) \models Tr^{bound}(D)$.

2. *Assume that* $a \in I_K$ *and the interval* $I_K \restriction < a$ *is not empty (i.e.* a *is not the least element of* I_K*). Then* $K \restriction \leq a \models D$ *iff* $(K,\ a) \models Tr^{ubound}(D)$.

3. *Assume that* $b \in I_K$ *and the interval* $I_K \restriction > b$ *is not empty (i.e.* b *is not the maximal element of* I_K*). Then* $K \restriction \geq b \models D$ *iff* $(K,\ b) \models Tr^{lbound}(D)$.

4. $K \models D$ *iff* $K \models Tr(D)$.

Proof. The proof proceeds by structural induction on PDC formulas. For the inductive step use Proposition 15 and Lemma 20. □
Finally, Theorem 17(1) follows from Lemma 21 and Lemma 20.

7 Proof of Theorem 17(2)

A finitely variable structure $K = \langle I, P_1^K, \ldots P_n^K \rangle$ is a *step structure* if I_K is a left closed right open subinterval of $\mathbb{R}^{\geq 0}$ and for every $a \in I_K$ and $i = 1, \ldots, n$ there is $b \in I_K$ such that $a < b$ and the predicate P_i^K is constant on the interval $[a,\ b)$. Let $STEP$ be the set of all step structures. It is clear that every finitely variable structure is $\sim_{fin}$-equivalent to a structure in $STEP$. Hence, if ϕ respects $\sim_{fin}$-equivalence and a duration formula D is equivalent to ϕ over all structures in $STEP$, then ϕ is equivalent to D over all finitely variable structures. Therefore, in order to prove Theorem 17(2) it is sufficient to provide an algorithm that for every sentence ϕ constructs a duration formula D such that ϕ is equivalent to D over all structures in $STEP$. It is easier to prove a stronger result (see Proposition 22 below). Namely, let $L(min, <)$ be the extension of monadic language of order by the constant symbol min. The

formulas of $L(min, <)$ will be interpreted over the structures in $STEP$; the constant min will be interpreted in $\langle I_K,\ <_K,\ P_1^K, \ldots, P_n^K\rangle$ as the minimal element min_K of I_K. We will show

Proposition 22 *There exists a translation algorithm Tr from sentences of $L(min, <)$ to PDC formulas such that ϕ is equivalent to $Tr(\phi)$ over all structures in $STEP$.*

Proof. Throughout the proof we use "equivalent" for "equivalent over $STEP$".

Observe that if D_1 is equivalent to φ_1 and D_2 is equivalent to φ_2, then $D_1 \vee D_2$ is equivalent to $\varphi_1 \vee \varphi_2$. Similar observations hold for conjunction and negation.

The proposition is proved by induction on the quantifier depth of the sentences. In view of the above observation, it is sufficient to prove the proposition for the atomic sentences and to carry out the inductive step for sentences of the form $\exists t.\ \phi(t)$.

Inductive Basis. The only atomic sentences of $L(min, <)$ are $min < min$ and sentences of the form $X_i(min)$. Let D be any DC formula and let $TRUE$ be the abbreviation for $D \vee \neg D$. We translate the sentence $min < min$ to $\neg TRUE$ and the sentence $X_i(min)$ to $\lceil X_i \rceil^\frown TRUE$. It is easy to verify that $min < min$ is equivalent to $\neg TRUE$ and $X_i(min)$ is equivalent (over $STEP$) to $\lceil X_i \rceil^\frown TRUE$. This completes the inductive basis.

Inductive Step. Assuming that every sentence of $L(min, <)$ of quantifier depth at most n is equivalent over $STEP$ to a PDC formula we are going to prove that for every formula $\varphi(t)$ of quantifier depth n, the sentence $\exists t.\ \varphi(t)$ is equivalent over $STEP$ to a PDC formula.

Observe that $\exists t.\varphi(t) \leftrightarrow (\varphi\{min/t\} \vee \exists t.t > min \wedge \varphi(t))$. The quantifier depth of $\varphi\{min/t\}$ is n, therefore, by the inductive assumption, it is equivalent to a PDC formula. Hence, it is sufficient to prove that $\exists t.t > min \wedge \varphi(t)$ is equivalent to a PDC formula.

Let t_1 and t_2 be two fresh variables. Let $\phi(t_1,\ t_2)$ be obtained from $\varphi(t_2)$ by replacing all the occurrences of min by t_1. It is clear that

$$K \models \varphi(e) \text{ iff } K \models \phi(min_K, e) \tag{1}$$

By Lemma 7 there are $\psi^i(t_1,\ t_2), \psi^i_{\geq}(t_2)$ such that for every $K \in STEP$ and for every $e > min_K$

$$K \models \phi(min_K,\ e) \text{ iff } K \models \bigvee_i (\psi^i(min_K,\ e) \wedge \psi^i_{\geq}(e)) \tag{2}$$

and the quantifier depths of $\psi^i(t_1,\ t_2)$ and $\psi^i_{\geq}(t_1)$ are at most n, and $\psi^i(t_1,\ t_2)$ is explicitly restricted to $[t_1,\ t_2)$ and $\psi^i_{\geq}(t_2)$ is explicitly restricted by t_2 from below.

Let $\chi^i(t_1)$ be obtained from $\psi^i(t_1,\ t_2)$ as in Lemma 4. Then

$$K \models \psi^i(min_K,\ e) \text{ iff } K \restriction [min_K, e) \models \chi^i(min_K) \tag{3}$$

Since $\chi^i\{min/t_1\}$ has the quantifier depth $\leq n$, there is D^i which is equivalent over $STEP$ to $\chi^i\{min/t_1\}$. In particular,

$$K \restriction [min_K, e) \models D^i \text{ iff } K \restriction [min_K, e) \models \chi^i(min_K) \tag{4}$$

According to Lemma 3, $K \models \psi^i_{\geq}(e)$ iff $K \restriction \geq e \models \psi^i_{\geq}(e)$. Observe that $K \restriction \geq e \in STEP$ (because I_K is right open), and e is the least element of $K \restriction \geq e$. The quantifier depth of $\psi^i_{\geq}(t_2)\{min/t_2\}$ is $\leq n$, therefore, by the inductive assumption there is a PDC formula $D^i_{\geq}$ which is equivalent over $STEP$ to $\psi^i_{\geq}(t_2)\{min/t_2\}$. In particular,

$$K \restriction \geq e \models \psi^i_{\geq}(e)) \text{ iff } K \restriction \geq e \models D^i_{\geq} \tag{5}$$

Recall that $K \models (D^i)^\frown D^i_{\geq}$ iff there is $e > min_K$ such that $K \restriction [min_K,\ e) \models D^i$ and $K \restriction \geq e \models D^i_{\geq}$. Therefore, from the equations (1)-(5) we obtain that $K \models \exists t.\ t > min \wedge \varphi(t)$ iff $K \models \bigvee_i ((D^i)^\frown D^i_{\geq})$. This completes the inductive step.

Finally, observe that all constructions in the above proof were recursive. □

Let us mention the following corollary of the above proof.

Corollary 23 *Let ϕ be a first-order monadic sentence. Assume that if $P_i,\ P'_i$ are finitely variable predicates over $\mathbb{R}^{\geq 0}$ such that $P_i \sim_{fin} P'_i$ for $i = 1, \ldots n$, then $\langle \mathbb{R}^{\geq 0}, <, P_1, \ldots, P_n\rangle \models \phi$ iff $\langle \mathbb{R}^{\geq 0}, <, P'_1, \ldots, P'_n\rangle \models \phi$. Then there is a PDC formula D such that for every finitely variable structure K over $\mathbb{R}^{\geq 0}$*

$$K \models \phi \text{ if and only if } K \models D.$$

Moreover D is computable from ϕ.

8 Durational Calculus and Liveness

In this section we compare the "traditional" extensions of the semantical definition for DC to infinite intervals with the definition used in our paper.

We show that liveness properties cannot be specified under the traditional definitions. We demonstrate that extensions of DC by new modalities suggested in the literature (see e.g. [10, 18]) to handle liveness can be easily embedded into PDC investigated in our paper.

In the DC literature the term "interpretation" (or "valuation") is used for a finitely variable structure over the set of all non-negative reals.

Throughout this section K will stand for such an interpretation $\langle \mathbb{R}^{\geq 0}, <, P_1, \ldots, P_n \rangle$.

The main semantical relation considered in the DC literature is "a formula D holds in a bounded closed interval $[a,\ b]$ under an interpretation K". Here we will use the notation $K,\ [a,\ b] \models_{bound} D$ for that relation; we will use the symbol $\models$ for both the satisfaction relation of first-order monadic logic of order and for the satisfaction relation for PDC as defined in Section 4.2.

Our semantics $\models$ for PDC is consistent with $\models_{bound}$ and extends it to infinite intervals. For the bounded intervals $[a, b]$ the relations $\models$ and $\models_{bound}$ are related as follows

$$K,\ [a, b] \models_{bound} D \text{ if and only if } K \upharpoonright [a, b] \models D.$$

8.1 The Standard Extension of DC to Infinite Intervals

Two standard extensions $\models^u$ and $\models^p$ of $\models_{bound}$ to the interval of all non-negative reals are defined as follows.

Definition 24 *(universal satisfaction $K \models^u D$ and prefix satisfaction $K \models^p D$)*

1. *$K \models^u D$ iff $K, [a_1, a_2] \models_{bound} D$ for all $a_1 < a_2$.*
2. *$K \models^p D$ iff $K, [0, a_2] \models_{bound} D$ for all $a_2 > 0$.*

The next proposition shows that the satisfaction relations $\models^p$ and $\models^u$ can be encoded by our satisfaction relation $\models$.

Proposition 25 *For every PDC formula D there are PDC formulas D^u and D^p such that*

1. *$K \models^u D$ iff $K \models D^u$.*
2. *$K \models^p D$ iff $K \models D^p$.*

Proof. Let $\phi(t_1, t_2)$ be defined as $Tr^{bound}(D)$ (see Figure 1 for the definition of Tr^{bound}). By Lemma 21, for $a_1 < a_2$

$$K \upharpoonright [a_1, a_2] \models D \text{ if and only if } (K, a_1, a_2) \models \phi(t_1, t_2)$$

Therefore, by the consistency of $\models_{bound}$ and $\models$, it follows that

$$K, [a_1, a_2] \models_{bound} D \text{ if and only if } (K, a_1, a_2) \models \phi(t_1, t_2)$$

Hence, by the definition of $\models^u$ and $\models^p$ we obtain that

$$K \models^u D \text{ if and only if } K \models \forall t_1 \forall t_2.\ t_2 > t_1 \rightarrow \phi(t_1, t_2)$$

and

$$K \models^p D \text{ if and only if } K \models \forall t_1 \forall t_2.\ Zero(t_1) \wedge t_2 > t_1 \rightarrow \phi(t_1, t_2),$$

where $Zero(t) \triangleq \forall t'.t \leq t'$.

By Corollary 23, there are PDC formulas D^u and D^p such that

$$K \models D^u \text{ if and only if } K \models \forall t_1 \forall t_2.\ t_2 > t_1 \rightarrow \phi(t_1, t_2)$$

and

$$K \models D^p \text{ if and only if } K \models \forall t_1 \forall t_2.\ Zero(t_1) \wedge t_2 > t_1 \rightarrow \phi(t_1, t_2)$$

Therefore, $K \models^u D$ iff $K \models D^u$ and $K \models^p D$ iff $K \models D^p$. □

The reader can also verify directly that

$$K \models^u D \text{ if and only if } K \models \neg(TRUE^\frown(\neg D)^\frown TRUE) \text{ and}$$

$$K \models^p D \text{ if and only if } K \models \neg((\neg D)^\frown TRUE),$$

where $TRUE$ is the abbreviation for $\lceil X \vee \neg X \rceil$.

8.2 Liveness

It is a widely believed misconception that the Duration Calculus cannot specify liveness properties while monadic first-order logic can. Theorem 17 shows that PDC (under $\models$ semantics) and first-order monadic logic of order have the same expressive power. Therefore, liveness properties can be specified in

PDC (under the extension $\models$ of $\models_{bound}$ to infinite intervals). The following are the sources of this misconception.

The "traditional" semantics $\models_{bound}$ of DC is defined only on the bounded closed subintervals of the reals. There are no liveness properties over bounded time domains, hence, the question whether $\models_{bound}$ can capture liveness is meaningless. The extensions $\models^p$ and $\models^u$ of DC to infinite intervals considered in the literature use a convention that a formula holds on an infinite interval if it holds on all its finite length prefixes (or on all its bounded subintervals). If such a convention is used, only safety properties can be expressed.

In this subsection we show that $\models^p$ and $\models^u$ cannot specify even a very simple liveness property. Hence, unlike the PDC semantics $\models$ suggested here, the semantics $\models^p$ and $\models^u$ are not expressively complete. Therefore, they are not appropriate extensions of $\models_{bound}$ to infinite intervals.

One of the simplest liveness properties that respects $\sim_{fin}$ equivalence is "Eventually there is a non-empty open subinterval where X holds". It is clear that this property can be formalized in first-order monadic logic. Therefore, by the expressive completeness of PDC (under our semantical definitions of $\models$), there is a PDC formula D which formalizes this property.

Let EV be the set of all interpretations K that have the above property. The reader can verify that $K \in EV$ iff $K \models \lceil X \vee \neg X \rceil^\frown \lceil X \rceil^\frown \lceil X \vee \neg X \rceil$. Let us show that this property cannot be formalized under the traditional PDC semantics $\models^u$ and $\models^p$.

Proposition 26 *1. There is no D such that $K \models^u D$ if and only if $K \in EV$.*

2. There is no D such that $K \models^p D$ if and only if $K \in EV$.

Proof. (1) Let $P^{(n)} = \{a \ : \ a > n\}$ and Let $P^\emptyset$ be the empty subset of $\mathbb{R}^{\geq 0}$. Let $K^{(n)}$ be the structure $\langle \mathbb{R}^{\geq 0}, <, \ P^{(n)} \rangle$ and let $K^\emptyset$ be $\langle \mathbb{R}^{\geq 0}, <, \ P^\emptyset \rangle$.

Observe that $K^{(n)}$ is in EV (for $n \in Nat$) and $K^\emptyset$ is not in EV.

Assume that $K^{(n)} \models^u D$ for every $n \in Nat$. Then, $K^{(n)}, [a_1, a_2] \models_{bound} D$ for every $a_1 < a_2$. Hence, $K^{(n)} \restriction [a_1, a_2] \models D$. However, $K^{(n)} \restriction [a_1, a_2]$ coincides with $K^\emptyset \restriction [a_1, a_2]$ for every $a_1 < a_2 < n$. Therefore, $K^\emptyset, [a_1, a_2] \models_{bound} D$ for every $a_1 < a_2$. Hence, $K^\emptyset \models^u D$.

From the assumption that $K^{(n)} \models^p D$ for every $n \in Nat$ we have derived that $K^\emptyset \models^u D$. Therefore, there is no D such that $K \in EV$ iff $K \models^u D$.

(2) The proof for $\models^p$ is almost the same. $\square$

8.3 New Modalities

We have seen that even a very simple liveness property cannot be specified in DC under the traditional extensions $\models^u$ and $\models^p$ of DC semantics to infinite intervals.

In order to capture liveness properties such as fairness and asymptotic stability a large number of "outward looking" modalities were suggested in the literature. All these extensions can be easily formalized in the fragment of first-order monadic logic that respects $\sim_{fin}$-equivalence. Hence, in view of Theorem 17, these new modalities can be encoded in the extension of DC considered here.

Below we describe an extension of DC by two new chopping operators suggested in [18]. This extension was called Duration Calculus with Liveness (DCL) and it was shown that in contrast to the original Duration Calculus, DCL can be used to specify some liveness and fairness properties (under $\models^p$ semantics). We show that DCL can be easily formalized in first order monadic logic and therefore, does not increase the expressive power of DC (under $\models$ semantics). Similar results hold for other extensions of DC proposed in the literature.

Two new chopping operators (modalities) of DCL are denoted by $\lhd$ and $\rhd$ and their semantics is defined as follows:

1. $K, [a_1, a_2] \models_{bound} D_1 \lhd D_2$ if there is $a_3 < a_1$ such that $K, [a_3, a_1] \models_{bound} D_1$ and $K, [a_3, a_2] \models_{bound} D_2$.

2. $K, [a_1, a_2] \models_{bound} D_1 \rhd D_2$ if there is $a > a_2$ such that $K, [a_1, a_3] \models_{bound} D_1$ and $K, [a_2, a_3] \models_{bound} D_2$.

It is clear that these modalities are first order definable. Let us extend Tr^{bound} (see Figure 1) to the DCL formulas as follows:

$$Tr^{bound}(D_1 \lhd D_2) \triangleq \exists t_3.\, t_3 < t_1 \wedge Tr^{bound}(D_1)\{t_3/t_1, t_1/t_2\} \wedge Tr^{bound}(D_2)\{t_3/t_1\}$$

$$Tr^{bound}(D_1 \rhd D_2) \triangleq \exists t_3.\, t_3 > t_2 \wedge Tr^{bound}(D_1)\{t_3/t_2\} \wedge Tr^{bound}(D_2)\{t_2/t_1, t_2/t_2\}$$

It is easy to check that for every DCL formula D its Tr^{bound} translation $\phi(t_1, t_2)$ has the following property

$$K, [a_1, a_2] \models_{bound} D \text{ if and only if } (K, a_1, a_2) \models Tr^{bound}(D).$$

However, in contrast to the Tr^{bound} translation of the PDC formulas, the Tr^{bound} translation of the DCL formulas are not explicitly restricted to $(t_1,\ t_2)$.

The relations $\models^u$ and $\models^p$ for DCL are defined from $\models_{bound}$ as in Definition 24. The same arguments as in the proof of Proposition 25 show

Proposition 27 *For every DCL formula D there are PDC formulas D^u and D^p such that*

1. *$K \models^u D$ iff $K \models D^u$.*
2. *$K \models^p D$ iff $K \models D^p$.*

Proof. Let D be a DCL formula and let $\phi(t_1, t_2)$ be $Tr^{bound}(D)$. Then

$$K, [a_1, a_2] \models_{bound} D \text{ if and only if } (K, a_1, a_2) \models \phi(t_1, t_2)$$

Hence, by the definition of $\models^u$ and $\models^p$ we obtain that

$$K \models^u D \text{ if and only if } K \models \forall t_1 \forall t_2.\ t_2 > t_1 \rightarrow \phi(t_1, t_2)$$

and

$$K \models^p D \text{ if and only if } K \models \forall t_1 \forall t_2.\ Zero(t_1) \wedge t_2 > t_1 \rightarrow \phi(t_1, t_2),$$

where $Zero(t) \stackrel{\Delta}{=} \forall t'.t \leq t'$.

By Corollary 23, there are PDC formulas D^u and D^p such that

$$K \models D^u \text{ if and only if } K \models \forall t_1 \forall t_2.\ t_2 > t_1 \rightarrow \phi(t_1, t_2)$$

and

$$K \models D^p \text{ if and only if } K \models \forall t_1 \forall t_2.\ Zero(t_1) \wedge t_2 > t_1 \rightarrow \phi(t_1, t_2)$$

Therefore, $K \models^u D$ iff $K \models D^u$ and $K \models^p D$ iff $K \models D^p$. □

9 Star Free Expressions and PDC

We provide a novel stuttering free interpretation for star-free expressions. The main result of this section establishes the equivalence between star-free expressions under stuttering free interpretation and PDC.

9.1 Star free expressions

The (extended) star free regular expressions over an alphabet Σ [7] are defined by the following grammar:

$$E ::= \ l \mid E + E \mid E; E \mid \neg E, \text{ where } l \text{ ranges over } \Sigma.$$

In this paper we will use "star free expression" for "extended star free regular expression".

The standard interpretation assigns to a star free expression a set of string (language) over Σ. In this interpretation sum ($+$) is interpreted as union, sequential composition ($;$) is interpreted as concatenation and negation ($\neg$) is interpreted as the complementation relative to the set of all finite strings (excluding the empty string ϵ).

We use $[\![E]\!]$ for the language assigned to the star free expression E under the standard interpretation. Expressions E^1 and E^2 are said to be equivalent (under the standard interpretation) if $[\![E^1]\!] = [\![E^2]\!]$.

9.2 Stuttering Free Interpretation

Definition 28 *(Stuttering [6]) A string $l_0 l_1 \ldots l_n$ is stuttering free iff it is non empty and no two adjacent symbols in it are equal.*

Let us consider stuttering free interpretations of negation and sequential composition symbols. Namely, let $\neg$ be the complementation relative to the set of stuttering free strings and let sequential composition be interpreted as the following operation $\star$ on strings:

$$l_0 \ldots l_p \star m_0 \ldots m_k = \begin{cases} l_0 \ldots l_p m_1 \ldots m_k & \text{if } l_p = m_0 \\ l_0 \ldots l_p m_0 \ldots m_k & \text{otherwise.} \end{cases}$$

Sum, like before, is interpreted as union. The stuttering free interpretation assigns to a star free expression E the stuttering free string language which will be denoted by $[\![E]\!]^{stut}$.

9.3 Finitely Variable Functions and their Traces

We say that a function η from a bounded subinterval $(a,\ b)$ of the reals into a finite set Σ has *finite variability* if there exists a finite increasing sequence $a = a_0 < a_1 < a_2 \ldots < a_n = b$ such that η is constant on every interval (a_i, a_{i+1}).

The restriction of η on an interval $(c,\ d)$ is denoted by $\eta \restriction (c,\ d)$. Notice that if $\eta :\ (a,\ b) \rightarrow \Sigma$ has finite variability and $(c,\ d) \subseteq (a,\ b)$, then $\eta \restriction (c,\ d)$ has finite variability.

The following lemma is straightforward.

Lemma 29 *Suppose that $\eta :\ (a,\ b) \rightarrow \Sigma$ has finite variability, then there exists a unique increasing sequence $a = a_0 < a_1 < a_2 \ldots < a_n = b$ such that*

1. *η is almost constant on every interval $(a_i,\ a_{i+1})$, i.e., for every i there is $l_i \in \Sigma$ such that the set $\{x \in (a_i,\ a_{i+1}) \ :\ \eta(x) \neq l_i\}$ is finite.*

2. *For every $i < n-1$, the value of η on (a_i, a_{i+1}) differs from the value of η on (a_{i+1}, a_{i+2}).*

Definition 30 *(Trace of a finite variability function.) Let η be a finite variability function over $(a,\ b)$ and let $a_0, \ldots, a_n$ be as in Lemma 29. For $i < n$ let l_i be the values of η on (a_i, a_{i+1}). The trace of η (notations* **trace**(η)*) is the stuttering free string $l_0 l_1 \ldots l_{n-1}$.*

Lemma 31 *Suppose $\eta :\ (a,\ b) \rightarrow \Sigma$ and $c \in (a,\ b)$. Then* **trace**(η)= **trace**$(\eta \restriction (a,\ c))\star$**trace**$(\eta \restriction (c,\ b))$*, where $\star$ is stuttering free concatenation (see subsection 9.2).*

Trace of a tuple: Let $\langle \eta_1, \ldots, \eta_n \rangle$ be an n-tuple of finite variability functions from $(a,\ b)$ into $\{0,\ 1\}$. With this n-tuple we associate a function η from $(a,\ b)$ into $\{0, 1, \ldots, 2^n - 1\}$ defined as $\eta(t) = i$ if $\langle \eta_1(t), \ldots, \eta_n(t) \rangle$ is the binary representation of i. The above mapping defines a one-to-one correspondence between the set of n-tuple of finite variability functions from $(a,\ b)$ into $\{0, 1\}$ and finite variability functions from $(a,\ b)$ into $\{0, \ldots, 2^n - 1\}$. The trace of an n-tuple (notations **trace**$(\eta_1, \ldots, \eta_n)$) is defined as the trace of the corresponding function.

Trace of a structure: Let $K = \langle I_K,\ <_K,\ P_1^K, \ldots, P_n^K \rangle$ be a finitely variable structure over a bounded interval I_K of the reals. Let $a = inf\{c \ : c \in I_K\}$ and $b = sup\{c \ : c \in I_K\}$ The trace of K (notations **trace**(K)) is defined as the trace of the n-tuple $\langle P_1^K \restriction (a,b), \ldots, P_n^K \restriction (a,b) \rangle$

9.4 Translations between Star Free Expressions and PDC

In this subsection we provide meaning preserving translations between the PDC formulas over the state variables $X_1, \ldots, X_n$ and the star free expressions over the alphabet $\{0, 1, \ldots 2^n - 1\}$.

Throughout this section, $K = \langle I_K, <_K, P_1^K, \ldots, P_n^K \rangle$ is a finitely variable structure with n unary predicates over a bounded interval I_K.

Let $i \in \{0, 1, \ldots 2^n - 1\}$ and let $a_1 \ldots a_n$ be the binary representation of i. For $j = 1, \ldots n$ define

$$Y_j^{(i)} \triangleq \begin{cases} X_j & \text{if } a_j = 1 \\ \neg X_j & \text{if } a_j = 0 \end{cases} \tag{6}$$

The following lemmas are immediate

Lemma 32 *$K \models \lceil Y_1^{(i)} \wedge Y_2^{(i)} \wedge \ldots Y_n^{(i)} \rceil$ if and only if* $\mathbf{trace}(K) = i$.

Lemma 33 *$K \models \lceil \neg Y_1^{(i)} \vee \neg Y_2^{(i)} \vee \ldots \vee \neg Y_n^{(i)} \rceil$ if and only if i does not occur in* $\mathbf{trace}(K)$.

Proposition 34 *1. For every $i \in \{0, 1, \ldots 2^n - 1\}$ there exists a state expression S_i over the variables $X_1, \ldots, X_n$ such that $K \models \lceil S_i \rceil$ if and only if* $\mathbf{trace}(K) = i$.

2. For every state expression S over the variables $X_1, \ldots, X_n$ there exists a star free expression E_S over the alphabet $\{0, 1, \ldots 2^n - 1\}$ such that $K \models \lceil S \rceil$ if and only if $\mathbf{trace}(K) \in [\![E_S]\!]^{stut}$.

Proof. Proposition 34(1) is obtained from Lemma 32. In order to prove Proposition 34(2), it is useful to introduce the following abbreviations.

1. $E_1 \cap E_2 \triangleq \neg(\neg E_1 + \neg E_2)$.
2. $\Sigma_{2^n} = 0 + 1 + 2 + \ldots + 2^n - 1$.
3. $\emptyset \triangleq 0 \cap \neg 0$.
4. $\Sigma_{2^n}^+ \triangleq \neg\emptyset$.
5. $\mathrm{Occur}(i) \triangleq i; \Sigma_{2^n}^+ + \Sigma_{2^n}^+; i; \Sigma_{2^n}^+ + \Sigma_{2^n}^+; i$.

It is easy to see that the language definable (under both the standard and the stuttering free interpretations) by $E_1 \cap E_2$ is the intersection of the languages definable by E_1 and E_2; the language definable by $\emptyset$ is the empty language; the language definable by $\Sigma_{2^n}^+$ consists of all stuttering free strings over Σ_{2^n}; the language definable by $Occur(i)$ under the stuttering free interpretation (respectively under the standard interpretation) consists of all stuttering free

strings (respectively all finite strings) which contain an occurrence of the letter i.

Let S be a boolean combination of $X_1, \ldots, X_n$. If S is a tautology, then the expression E_S defined as $\Sigma^+_{2^n}$ satisfies Proposition 34(2). Let us assume that S is not a tautology. In this case there is a nonempty set $I \subseteq \{0, 1, \ldots 2^n - 1\}$ such that $S \leftrightarrow \wedge_{i \in I}(\neg Y_1^{(i)} \vee \neg Y_2^{(i)} \vee \ldots \vee \neg Y_n^{(i)})$. This set I is easily obtained from the complete conjunctive normal form of S. Therefore, $K \models \lceil S \rceil$ iff $K \models \lceil \wedge_{i \in I}(\neg Y_1^{(i)} \vee \neg Y_2^{(i)} \vee \ldots \vee \neg Y_n^{(i)}) \rceil$ iff $K \models \wedge_{i \in I} \lceil \neg Y_1^{(i)} \vee \neg Y_2^{(i)} \vee \ldots \vee \neg Y_n^{(i)} \rceil$. Hence, by Lemma 33, it follows that $K \models \lceil S \rceil$ iff no i from I occurs in the $\mathbf{trace}(K)$. Therefore, the star free expression E_S defined as $\bigcap_{i \in I} \neg Occur(i)$ satisfies Proposition 34(2). □

Below we are going to define translations Tr_n and Tr'_n between the PDC formulas over the state variables $X_1, \ldots, X_n$ and the star free expressions over the alphabet $\{0, 1, \ldots 2^n - 1\}$.

Let S_i be as in Proposition 34(1). Define the translation Tr_n from the star free expressions over the alphabet $\{0, 1, \ldots, 2^n - 1\}$ into PDC formula over the variables $X_1, \ldots, X_n$ as follows:

1. $Tr_n(i) \triangleq \lceil S_i \rceil$.
2. $Tr_n(E_1 + E_2) \triangleq Tr_n(E_1) \vee Tr_n(E_2)$.
3. $Tr_n(\neg E) \triangleq \neg Tr_n(E)$.
4. $Tr_n(E_1; E_2) \triangleq Tr_n(E_1) \frown Tr_n(E_2)$.

Let E_S be as in Proposition 34(2). Define the translation Tr'_n from the PDC formula over the variables $X_1, \ldots, X_n$ into the star free expressions over the alphabet $\{0, 1, \ldots, 2^n - 1\}$ as follows:

1. $Tr'_n(\lceil S \rceil) \triangleq E_S$.
2. $Tr_n(D_1 \vee D_2) \triangleq Tr'_n(D_1) + Tr'_n(D_2)$.
3. $Tr'_n(\neg D) \triangleq \neg Tr'_n(D)$.
4. $Tr'_n(D_1 \frown D_2) \triangleq Tr'_n(D_1); Tr'_n(D_2)$.

The main result of this section is the following equivalence between the star-free expressions under stuttering free interpretation and PDC.

Theorem 35 *For every finitely variable structure* $K = \langle I_K, <_K, P_1^K, \ldots, P_n^K \rangle$ *over a bounded interval*

1. $K \models Tr_n(E)$ *iff* $\mathbf{trace}(K) \in [\![E]\!]^{stut}$.
2. $K \models D$ *iff* $\mathbf{trace}(K) \in [\![Tr'_n(D)]\!]^{stut}$.

Proof. (1) The proof proceeds by structural induction on star free expressions. For the inductive basis use Proposition 34. The inductive steps for $+$ and $\neg$ are immediate. The inductive step for concatenation follows from Lemma 31.

(2) The proof proceeds by structural induction on PDC formulas. For the inductive basis use Proposition 34. The inductive steps for the boolean connectives are immediate. The inductive step for chop follows from Lemma 31. □

Corollary 36 *(Equivalence between PDC and star-free expressions)*

1. $[\![E_1]\!]^{stut} = [\![E_2]\!]^{stut}$ *if and only if the PDC formulas* $Tr_n(E_1)$ *and* $Tr_n(E_2)$ *are equivalent on the finitely variable structures over bounded intervals.*
2. *PDC formulas* D_1 *and* D_2 *are equivalent on the finitely variable structures over bounded intervals if and only if* $[\![Tr'_n(D_1)]\!]^{stut} = [\![Tr'_n(D_2)]\!]^{stut}$.

It is only for the sake of a lighter notation that we restricted the results of this section to the bounded intervals; there is an obvious generalization to the unbounded intervals. In this more general case a star free expression is interpreted as a set of stuttering free strings and stuttering free ω-strings. The negation is interpreted as the complementation relative to the set of stuttering free strings and stuttering free ω-strings.

10 Conclusion and Related Results

In this paper the Propositional Duration Calculus was compared to monadic first order logic of order - a very fundamental formalism. Our main result shows that there exist meaning (semantics) preserving translations between PDC formulas and monadic sentences that respect $\sim_{fin}$-equivalence. This result confirms that the Duration Calculus is not an ad-hoc formalism. The definitions of the semantics for PDC given in this paper conservatively extend

the traditional semantics of PDC to all intervals. In particular, the expressive completeness theorem holds for both bounded and unbounded intervals.

There is a tight connection between PDC and star free expressions. The chop operator of PDC corresponds to concatenation. We provided a non-standard stuttering free interpretation for star free expressions and established the equivalence between star free expressions under the stuttering free interpretation and PDC.

10.1 Complexity of Translations

The translation from PDC to monadic logic is compositional and has linear time complexity. The upper bound for the complexity of our translation algorithm from monadic logic to PDC can be extracted from a careful analysis of the proofs of Lemma 5 and Proposition 22. The best upper bound which we were able to extract is non-elementary. Recall that a function F is non-elementary if there is no m such that $F(n)$ is less than $exp_m(n)$ for all n, where $exp_m(k)$ is the m-times iterated exponential function (e.g., $exp_2(k) = 2^{2^k}$).

The translations between PDC and star-free expressions are compositional. It is easy to check that for every n, the translations Tr_n and Tr'_n have linear complexity.

10.2 Succinctness

In [14] we proved that there is at least an exponential blow-up in every meaning preserving translation from monadic logic to PDC. Hence, there exists an exponential gap between the succinctness of monadic logic and that of duration calculus. The space complexity of the validity problems both for first order monadic logic and for PDC has a non-elementary lower bound [20, 13]. We believe that the succinctness gap between monadic logic and PDC is much higher than exponential and in particular there is no exponential space meaning preserving translation algorithm from monadic logic to PDC.

10.3 Mean Value Calculus

Recall that in DC it is impossible to specify instantaneous events. In [24], DC was extended to Mean Value Calculus in order to handle instantaneous events. In [11] we established that there exist meaning (semantics) preserving

translations between the Propositional fragment of Mean Value Calculus and first-order monadic logic.

10.4 Decidability

Our main result, Theorem 17, deals with definability in different formalisms. This is completely orthogonal to the decidability issues. The satisfiability problem for PDC was proved to be decidable in [23] by automata theoretical methods. The satisfiability (and equivalence) problem for PDC can be also reduced to the emptiness problem for star-free expressions.

However, there is a much simpler way to show the decidability of PDC and of many other much stronger formalisms. In [15] we have considered an interpretation of monadic second-order logic of order in the second-order structures of finitely variable signals. This logic allows quantifications over the finitely variable monadic predicates. We proved the decidability of monadic second-order logic under the finitely variable interpretation. The decidability of PDC is an immediate by-product of the decidability of monadic second-order logic.

It is not difficult to assign to every first-order monadic sentence ϕ a second-order monadic sentence ψ such that ϕ respects $\sim_{fin}$ equivalence if and only if ψ is satisfiable under the finitely variable interpretation of its free and bound monadic predicate variables. Therefore, as a consequence of the decidability of monadic second-order logic, one can obtain that the set of first-order monadic sentences that respect $\sim_{fin}$ equivalence is decidable.

10.5 The Kamp and the McNaughton-Papert Theorems

It is instructive to compare our completeness result with Kamp's theorem [4, 2, 1] which states that every monadic formula $\psi(t)$ with one free variable t is equivalent to a propositional temporal logic formula D. Our theorem states that every monadic sentence (formula without free variables) that respects $\sim_{fin}$-equivalence is equivalent to a PDC formula.

The McNaughton-Papert theorem [7] is close both conceptually and technically to our completeness theorem. This theorem establishes equivalence between first-order monadic logic over finite linear orders and star free expressions. Recall that the star free expressions over alphabet Σ are constructed from the elements of Σ by boolean operations and concatenation.

The meaning of a star free expression is a set of strings (language). A string w of length $m > 0$ over an alphabet $\{0,\ 1\}^n$ can be considered as a structure $K^w = \langle\{1, \ldots, m\},\ <,\ P_1^w, \ldots, P_n^w\rangle$, where $<$ is the standard order relation over $\{1, \ldots, m\}$ and $P_i^w(j) = 1$ iff the i-th components of the j-th letter of w is 1. A sentence ϕ with predicates $X_1, \ldots, X_n$ defines a set of strings (language) L_ϕ over the alphabet $\{0,\ 1\}^n$ that satisfies ϕ. The McNaughton Papert theorem states that a language over an alphabet $\{0,\ 1\}^n$ is definable by a star free expression if and only if it is definable by a first-order monadic sentence [7]. This theorem was extended to ω-languages in [5, 21].

In [12] an interpretation of star free expressions over the reals is provided. The expressive power of star free expressions is compared to the expressive power of first-order monadic logic of order over the reals. It is proved that these formalisms have the same expressive power under the interpretation of the monadic predicate symbols by arbitrary (no finite variability restriction) subsets of reals. This result provides a generalization of the classical McNaughton Papert theorem [7] from the finite orders to the order of the reals.

Acknowledgment

I would like to thank Alexander Dikovsky and the anonymous referees for their helpful suggestions.

References

[1] D. Gabbay, I. Hodkinson and M. Reynolds. Temporal Logic. Oxford Un. Press, 1994.

[2] D. Gabbay, A. Pnueli, S. Shelah and J. Stavi. On the temporal analysis of fairness. In *7th ACM Symp. on Principles of Prog. Lang.*, pp. 163-173, 1980.

[3] M. Hansen and Zhou Chaochen. Duration Calculus: Logical Foundations. In *Formal Aspects of Computing*, 9:283-330, 1997.

[4] H. Kamp. *Tense logic and the theory of linear order*. Ph.D. Thesis, Un. of California at LA, 1968.

[5] R. E. Ladner Application of model theoretical games to linear orders and finite automata theory. Information and Control 9, pp. 521-530, 1977.

[6] L. Lamport. The Temporal Logic of Actions. ACM Transactions on Programming Languages and Systems, 16(3), pp. 872-923, 1994.

[7] R. McNaughton and S. Papert. Counter-free automata. The MIT Press, 1971.

[8] J. Monk. Mathematical Logic. Springer-Verlag, 1976.

[9] B. Moszkowski. Reasoning about Digital Circuits. Ph. D. Thesis, Stanford, 1983.

[10] P. Pandya. Some extensions to Propositional Mean Value Calculus: Expressiveness and Decidability. In *Proceedings of Computer Science Logic 1995*, LNCS vol. 1092, 1995.

[11] A. Rabinovich. On Expressive Completeness of Duration and Mean Value Calculi. Proceedings of Express 97. In the Electronic Notes of Theoretical Computer Science vol 7, 1997.

[12] A. Rabinovich. Star Free expressions over the reals. *To appear in Theoretical Computer Science.*

[13] A. Rabinovich. Non-elementary Lower Bound for Propositional Duration Calculus. Information Processing Letters 66 (1998), 7-11.

[14] A. Rabinovich. Succinctness Gap between Monadic Logic and Duration Calculus. Technical Report, Tel Aviv University, 1997.

[15] A. Rabinovich. A Universal Continuous Time Specification Formalism. To appear in Journal of Logic and Computation.

[16] A. Ravn. Design of Embedded Real Time Computing Systems, Technical Report, Technical University of Denmark, 1995.

[17] A. Ravn, H. Richel and K. Hansen. Specifying and verifying requirement of real time systems. *IEEE Transaction on Software Eng.*, 1993.

[18] J. Skakkæbak. Liveness and Fairness in Duration Calculus. In Concur94, LNCS vol. 836, pp. 283-298,1994.

[19] J. Skakkæbak, A. Ravn, H. Richel, Zhou Chaochen. Specification of Embedded Real time Systems. In Proc. of 1992 Euromicro workshop on Real Time Systems. IEEE Computer Society Press.

[20] L. Stockmeyer. The complexity of decision problems in automata and logic, Ph.D. Thesis, MIT, 1974.

[21] W. Thomas. Star Free regular sets of ω-sequences. Information and Control 42, pp. 148-156, 1979.

[22] Zhou Chaochen, C.A.R. Hoare and A. Ravn. A calculus of Duration. *Information processing Letters,* 40(5):269-279, 1991.

[23] Zhou Chaochen, M. R. Hansen and P. Sestoft. Decidability and undecidablity results for Duration Calculus. In *STACS'93,* Lect. Notes in Comp. Sci. vol 665, pages 58-68, 1993.

[24] Zhou Chaochen and L. Xiaoshan. A mean value calculus of duration. In *A Classical Mind: Essays in Honor of C. A. R. Hoare,* pages 431-451, Prentice Hall, 1994.

[25] Zhou Chaochen, D. Hung and L. Xiaoshan. A duration Calculus with Infinite Intervals. In Fundamentals of Computation Theory, LNCS vol. 965, pp. 16-41, 1995.